UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P O Box 1450
Alexandria, Virginia 22313-1450
www uspto gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/092,010	03/05/2002	Eric D. Bloch	LZLO-01001US0	7583

28554 7590 08/17/2009
Vierra Magen Marcus & DeNiro LLP
575 Market Street, Suite 2500
San Francisco, CA 94105

EXAMINER	
AILES, BENJAMIN A	

ART UNIT	PAPER NUMBER
2442	

MAIL DATE	DELIVERY MODE
08/17/2009	PAPER

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

Office Action Summary	Application No.	Applicant(s)	
	10/092,010	BLOCH ET AL.	
	Examiner	Art Unit	
	BENJAMIN AILES	2442	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE *3* MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *18 May 2009*.

2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *See Continuation Sheet* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *See Continuation Sheet* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)	4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____ .
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)	5) ☐ Notice of Informal Patent Application
3) ☐ Information Disclosure Statement(s) (PTO/SB/08) Paper No(s)/Mail Date _____.	6) ☐ Other: _____.

Continuation of Disposition of Claims: Claims pending in the application are 1,4,5,7,8,11,13,21,22,28,30,31,33,41,43-45,47,51-58,60-62,64,65,67-70,73 and 77-102.

Continuation of Disposition of Claims: Claims rejected are 1,4,5,7,8,11,13,21,22,28,30,31,33,41,43-45,47,51-58,60-62,64,65,67-70,73 and 77-102.

DETAILED ACTION

Continued Examination Under 37 CFR 1.114

1. A request for continued examination under 37 CFR 1.114, including the fee set forth in 37 CFR 1.17(e), was filed in this application after final rejection. Since this application is eligible for continued examination under 37 CFR 1.114, and the fee set forth in 37 CFR 1.17(e) has been timely paid, the finality of the previous Office action has been withdrawn pursuant to 37 CFR 1.114. Applicant's submission filed on 18 May 2009 has been entered.

2. Claims 1, 4, 5, 7, 8, 11, 13, 21, 22, 28, 30, 31, 33, 41, 43-45, 47, 51-58, 60-62, 64, 65, 67-70, 73 and 77-102 remain pending.

Response to Amendment

3. Applicant's cancellation of claims 20, 27, and 36 renders the prior rejection of claims 20, 27 and 36 under 35 USC 112, second paragraph, moot and therefore the rejection has been withdrawn. Applicant's amendment to claim 47 overcomes the prior rejection of claim 47 under 35 USC 112, second paragraph, and therefore the rejection has been withdrawn.

Claim Rejections - 35 USC § 103

4. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains.

> Patentability shall not be negatived by the manner in which the invention was made.

5. This application currently names joint inventors. In considering patentability of the claims under 35 U.S.C. 103(a), the examiner presumes that the subject matter of the various claims was commonly owned at the time any inventions covered therein were made absent any evidence to the contrary. Applicant is advised of the obligation under 37 CFR 1.56 to point out the inventor and invention dates of each claim that was not commonly owned at the time a later invention was made in order for the examiner to consider the applicability of 35 U.S.C. 103(c) and potential 35 U.S.C. 102(e), (f) or (g) prior art under 35 U.S.C. 103(a).

6. Claims 1, 13, 21, 22, 45, 47, 53, 54, 67, 69, 70, 83-88 and 102 are rejected under 35 U.S.C. 103(a) as being unpatentable over Tuli (US 7,068,381 B1) in view of Davis et al. (US 6,643,696 B2), hereinafter referred to as Davis, and further in view of Dove et al. (US 2008/0034121 A1), hereinafter referred to as Dove.

7. Regarding claim 1, Tuli teaches a method for providing content, comprising the steps of:

receiving a request from a user device (col. 2, ll. 5-13, receive request at web server) for particular content, said request is received at a server (col. 2, ll. 5-13, receive request at web server);

accessing a mark-up language description of said particular content (col. 2, ll. 9-11, retrieve HTML), said mark-up language description includes one or more source files which describe behavior of said particular content on a user interface of said user device based on user interactions with the particular content via the user interface (col. 2, ll. 11-12);

compiling said mark-up language description of said particular content (col. 2, ll. 11-12, browser translator), including said data, to create executable code for said user device, said step of compiling is performed at said server in response to said request (col. 2, ll. 5-13, fulfill request at web server); and

transmitting said executable code and said data from said server to said user device for execution by (col. 2, ll. 40-42, data is sent back to user device) said user device to allow a user to access said particular content and said data via said user interface and according to said behavior and said user interactions (col. 2, ll. 40-42, data is processed at user device).

Tuli teaches the accessing of a mark-up language description for the displaying of graphics and text (col. 2, ll. 1-14) but not does not explicitly teach (a) “mark-up language description includes a reference to a view instance element, the view instance element includes a reference to data for rendering on said user interface” and “accessing said data at said external data source based on said one or more source files which define said connection to said external data source, said server performs said accessing." Tuli teaches the compilation of mark-up language for viewing by portable devices utilizing a Browser Translator (col. 2, ll. 10-12) which includes the information with respect to a view instance element that includes the tag name and attributes (col. 2, ll. 12-20, graphics and text of a normal web page) but does not specifically recite (b) the steps: “accessing source code for the view instance element, the source code includes a tag name and attributes; creating a script instruction to call an instantiation view function in a function call; adding the tag name and attributes to the function call as parameters, the instantiation view function determines, based on the parameters, what kind of objects the user device creates at a runtime of the executable code at the user device; and compiling the function call to byte code."

(a) In related art, Davis teaches the utilization of a view instance element including a data reference for rendering information as claimed wherein Davis teaches the usage of tags that reference specific content (i.e. image data) that is stored on an external server. Davis teaches on the aspect of accessing data at an external data source wherein a client device can send a request to a server for secondary content (col. 5, lines 54-58) and that the secondary content can be from an external data source (abstract, line 7). Tuli and Davis are analogous art because they are both from the same field of endeavor of computer systems. At the time of invention, it would have been obvious to one of ordinary skill in the art that Davis's method of calling an application from a previously downloaded webpage could be used with Tuli's method of compiling code at a server rather than at the client. After Davis's webpage is downloaded with Tuli's system, Davis's webpage would call the secondary application and Tuli's system would then proceed to locate and compile that secondary application for presentation to the client. The motivation for doing so would have been to allow the users of Tuli's system to be able to utilize content of the type described in Davis on a thin-client device (col. 1, ll. 14-18). Therefore it would have been obvious to combine Davis with Tuli for the benefit of utilizing more complex content on a thin-client device.

(b) As mentioned above, Tuli teaches the compilation of mark-up language for viewing by portable devices utilizing a Browser Translator (col. 2, ll. 10-12) which includes the information with respect to a view instance element that includes the tag name and attributes (col. 2, ll. 12-20, graphics and text of a normal web page). In related art, Dove teaches the compilation of data for creating a script instruction to call an instantiation view for a function call including parameters so that the user device creates at a runtime of the executable code, compiling the function call to byte code in at least p. 3, para. 0022 and 0023. Dove teaches the creation of graphical programs that are converted to an executable format wherein the graphical programs may be represented as a plurality of data structures that define or specify the operation of the respective graphical programs. The executable format (e.g. machine language code or an interpretable script or other similar executable format) can then be executed by a portable computing device. One of ordinary skill in the art at the time of the applicant's invention would have found it obvious to implement the integration of creating an executable format file including specific function calls for graphical features to be run by a portable computing device as taught by Dove in combination with the Browser Translator taught by Tuli. One of ordinary skill would have been motivated to combine Dove with Tuli to account for the reduced processor and memory capabilities that limit the computing power of portable computing devices (Dove, p. 2, para. 0016).

8. Regarding claim 13, Tuli teaches wherein said particular content includes a first application which runs on said user device after said executable code is transmitted from said server to said user device, the method further comprising the steps of: that

said particular content includes a first application (Tuli, col. 2, lines 5-13), and the steps of accessing a mark-up language description of content (Tuli, col. 2, lines 9-13), compiling said mark-up language description of content (Tuli, col. 2, lines 10-13), and transmitting said compiled mark-up language description of content to said client (Tuli, col. 2, ll. 40-42, data is sent back to user device). Tuli does not expressly teach the step of receiving a request from a client associated with said rendering entity for second content and that said second content includes a second application called by said first application. Davis teaches that a client device can send a request to a server for secondary content and that the second content can include a second application that is called by the first application (col. 5, lines 54-58). At the time of invention, it would have been obvious to one of ordinary skill in the art that Davis's method of calling an application from a previously downloaded webpage could be used with Tuli's method of compiling code at a server rather than at the client. After Davis's webpage is downloaded with Tuli's system, Davis's webpage would call the secondary application and Tuli's system would then proceed to locate and compile that secondary application for presentation to the client. The motivation for doing so would have been to allow the users of Tuli's system to be able to utilize content of the type described in Davis on a thin-client device (col. 1, lines 59-61). Therefore it would have been obvious to combine Davis with Tuli for the benefit of utilizing more complex content on a thin-client device to obtain the invention as specified in claim 13.

9. Regarding claim 21, Tuli teaches a method for providing content, comprising the steps of:

receiving a request for content that includes data other than code (col. 2, ll. 5-13, receive request at web server), said data is for rendering on a user interface at a client, and said request is received at a server (col. 2, ll. 5-13, receive request at web server);

accessing a mark-up language description associated with said content at said server (col. 2, ll. 9-11, retrieve HTML);

acquiring said data from a data source external to and different than said server in response to said mark-up language description, said data is acquired by said server (col. 2, lines 46-47);

compiling said content at said server to create executable code, said content is based on said mark-up language description and said data, said executable code includes a representation of said data, said step of compiling is performed in response to said request (col. 2, ll. 11-12, browser translator); and

transmitting said executable code from said server to a client (col. 2, ll. 40-42, data is sent back to user device), said executable code implements said user interface, said user interface allows a user to access and dynamically interact with said data (col. 3, ll. 20-25).

Tuli does not expressly disclose the step of receiving a request from said client for second content and that said second content includes a second application called by said first application. Davis teaches that a client device can send a request to a server for secondary content and that the second content can include a second application that is called by the first application (col. 5, lines 54-58). At the time of invention, it would have been obvious to one of ordinary skill in the art that Davis's method of calling an application from a previously downloaded webpage could be used with Tuli's method of compiling code at a server rather than at the client. After Davis's webpage is downloaded with Tuli's system, Davis's webpage would call the secondary application and Tuli's system would then proceed to locate and compile that secondary application for presentation to the client. The motivation for doing so would have been to allow the users of Tuli's system to be able to utilize content of the type described in Davis on a thin-client device (col. 1, lines 59-61). Therefore it would have been obvious to combine Davis with Tuli for the benefit of utilizing more complex content on a thin-client device to obtain the invention as specified in claim 21.

10. Regarding claim 22, Tuli and Davis teach that said request is from said client (Tuli, col. 2, ll. 9-10).

11. Regarding claim 45, Tuli teaches an apparatus, comprising:

one or more storage devices (col. 2, ll. 6, host computer); and

one or more processors in communication with said one or more storage devices (col. 2, ll. 6, host computer), said one or more processors perform a method comprising the steps of:

receiving a request for particular content, said request is received at a server (col. 2, ll. 5-13, receive request at web server), said request is from a client, said client includes a browser and a rendering engine that is different than said browser but operates in connection with said browser (col. 2, ll. 21-25);

accessing a mark-up language description of said particular content (col. 2, ll. 9-11, retrieve HTML), said mark-up language description includes one or more source files which describe behavior of said particular content on a user interface of said user device (col. 2, ll. 11-12), said particular content includes data for rendering on said user interface (col. 2, ll. 1-14, graphics and text);

compiling said mark-up language description of said particular content (col. 2, ll. 11-12, browser translator), including said data, to create executable code for said user device, said step of compiling is performed at said server in response to said request (col. 2, ll. 5-13, fulfill request at web server); and

transmitting said executable code from said server to said user device (col. 2, ll. 40-42, data is sent back to user device), said user device provides said particular content via said user interface according to said one or more source files when said user device executes said executable code (col. 2, ll. 40-42, data is processed at user device).

Tuli teaches the accessing of a mark-up language description for the displaying of graphics and text (col. 2, ll. 1-14) but not does not explicitly teach (a) “mark-up language description includes a reference to a view instance element, the view instance element includes a reference to data for rendering on said user interface” and “accessing said data at said external data source based on said one or more source files which define said connection to said external data source, said server performs said accessing." Tuli teaches the compilation of mark-up language for viewing by portable devices utilizing a Browser Translator (col. 2, ll. 10-12) which includes the information with respect to a view instance element that includes the tag name and attributes (col. 2, ll. 12-20, graphics and text of a normal web page) but does not specifically recite (b) the steps: “accessing source code for the view instance element, the source code includes a tag name and attributes; creating a script instruction to call an instantiation view function in a function call; adding the tag name and attributes to the function call as parameters, the instantiation view function determines, based on the parameters, what kind of objects the user device creates at a runtime of the executable code at the user device; and compiling the function call to byte code.”

(a) In related art, Davis teaches the utilization of a view instance element including a data reference for rendering information as claimed wherein Davis teaches the usage of tags that reference specific content (i.e. image data) that is stored on an external server. Davis teaches on the aspect of accessing data at an external data source wherein a client device can send a request to a server for secondary content (col. 5, lines 54-58) and that the secondary content can be from an external data source (abstract, line 7). Tuli and Davis are analogous art because they are both from the same field of endeavor of computer systems. At the time of invention, it would have been obvious to one of ordinary skill in the art that Davis’s method of calling an application from a previously downloaded webpage could be used with Tuli’s method of compiling code at a server rather than at the client. After Davis’s webpage is downloaded with Tuli’s system, Davis’s webpage would call the secondary application and Tuli’s system would then proceed to locate and compile that secondary application for presentation to the client. The motivation for doing so would have been to allow the users of Tuli’s system to be able to utilize content of the type described in Davis on a thin-client device (col. 1, ll. 14-18). Therefore it would have been obvious to combine Davis with Tuli for the benefit of utilizing more complex content on a thin-client device.

(b) As mentioned above, Tuli teaches the compilation of mark-up language for viewing by portable devices utilizing a Browser Translator (col. 2, ll. 10-12) which includes the information with respect to a view instance element that includes the tag name and attributes (col. 2, ll. 12-20, graphics and text of a normal web page). In related art, Dove teaches the compilation of data for creating a script instruction to call an instantiation view for a function call including parameters so that the user device creates at a runtime of the executable code, compiling the function call to byte code in at least p. 3, para. 0022 and 0023. Dove teaches the creation of graphical programs that are converted to an executable format wherein the graphical programs may be represented as a plurality of data structures that define or specify the operation of the respective graphical programs. The executable format (e.g. machine language code or an interpretable script or other similar executable format) can then be executed by a portable computing device. One of ordinary skill in the art at the time of the applicant's invention would have found it obvious to implement the integration of creating an executable format file including specific function calls for graphical features to be run by a portable computing device as taught by Dove in combination with the Browser Translator taught by Tuli. One of ordinary skill would have been motivated to combine Dove with Tuli to account for the reduced processor and memory capabilities that limit the computing power of portable computing devices (Dove, p. 2, para. 0016).

12. Regarding claim 47, Tuli, Davis and Dove teach wherein the one or more processors implement a media transcoder (Tuli, col. 2, ll. 9-11), said media transcoder transcodes said media content to an accepted format before the media content is added to the tag header, said transcoding is separate from said compiling of the first code (Tuli, col. 2, ll. 25-29).

13. Regarding claim 53, Tuli, Davis and Dove teach the method wherein said executable code comprises one or more binary files (Tuli, col. 4, ll. 18-21).

14. Regarding claim 54, Tuli, Davis and Dove teach the method wherein said executable code comprises at least one of object code and byte code (Tuli, col. 4, ll. 18-21).

15. Regarding claim 67, Tuli, Davis and Dove teach the method wherein:

said markup language description comprises elements which are identified by markup language tags (Tuli, col. 2, ll. 10-11, HTML); and

said elements comprise at least one element which references said connection to said external data source (Davis, col. 5, lines 54-58).

16. Regarding claim 69, Tuli, Davis and Dove teach a method wherein said compiling comprises parsing said markup language description to identify first and second types of elements in the markup language description, providing said first type of element to a first compiling module which is appropriate for said first type of element to obtain first object code, providing said second type of element to a second compiling module which is appropriate for said second type of element to obtain second object code, and assembling said first and second object code into a single executable, and said

transmitting said executable code comprises transmitting said single executable to said user device (Tuli, col. 2, ll. 9-13, browser translator).

17. Regarding claim 70, Tuli teaches the method wherein said first type of element provides a script which defines behavior (col. 4, ll. 16-22). Tuli teaches the accessing of a mark-up language description but not does not explicitly teach the definition of a connection to an external data source for data wherein the external data source is external to the server. However, in related art, Davis teaches on this aspect wherein a client device can send a request to a server for secondary content (col. 5, lines 54-58) and that the secondary content can be from an external data source (abstract, line 7). Tuli and Davis are analogous art because they are both from the same field of endeavor of computer systems. At the time of invention, it would have been obvious to one of ordinary skill in the art that Davis's method of calling an application from a previously downloaded webpage could be used with Tuli's method of compiling code at a server rather than at the client. After Davis's webpage is downloaded with Tuli's system, Davis's webpage would call the secondary application and Tuli's system would then proceed to locate and compile that secondary application for presentation to the client. The motivation for doing so would have been to allow the users of Tuli's system to be able to utilize content of the type described in Davis on a thin-client device (col. 1, lines 59-61). Therefore it would have been obvious to combine Davis with Tuli for the benefit of utilizing more complex content on a thin-client device.

18. Regarding claim 83, Tuli and Davis teach the method wherein said executable code provides a script which is executed when a specified event occurs when a user

interacts with the particular content via the user interface, the specified event is based on user control of a pointing device or a key press (Tuli, col. 2, ll. 11-12).

19. Regarding claim 84, Tuli, Davis and Dove teach wherein the executable code, when executed at the user device, creates objects which are displayed on the user interface, the objects are created based on the instantiation view function, the tag name and the attributes (Dove, p. 3, para. 0022 and 0023).

20. Regarding claim 85, Tuli, Davis and Dove teach the method wherein the executable code, when executed at the user device, calls a predefined instantiation function associated with the tag name, and passes the attributes to the predefined instantiation function (Dove, p. 3, para. 0022 and 0023).

21. Regarding claim 86, Tuli, Davis and Dove teach the method wherein the executable code, when executed at the user device, calls a user-defined instantiation function associated with the tag name, and passes the attributes to the user-defined instantiation function (Dove, p. 3, para. 0022 and 0023).

22. Regarding claim 87, Tuli, Davis and Dove teach wherein the view instance element includes a reference to media content, and the one or more processors: access the media content (Dove, p. 3, para. 0022 and 0023); create an object representation of the media content, the object representation includes the media content and fields which store attributes of the media content, the attributes includes a name of the media content and a format of the media content (Dove, p. 3, para. 0022 and 0023); remove the media content from the object representation and insert a reference to the media content into the object representation in place of the media content, then compile the

object representation to byte code (Dove, p. 3, para. 0022 and 0023); create a tag header; add the media content, but not the compiled object representation, to the tag header, and transmit the compiled object representation with the compiled mark-up language description from said server to said user device for execution by said user device to provide said particular content and said data via said user interface according to said behavior and said user interactions (Dove, p. 3, para. 0022 and 0023).

23. Regarding claim 88, Tuli, Davis and Dove teach the method of assembling the compiled mark-up language description and the compiled object representation into a single executable which is transmitted as said executable code from said server to said user device (Dove, p. 3, para. 0022).

24. Regarding claim 102, Tuli, Davis and Dove teach wherein the view instance element includes a reference to media content, and the one or more processors: access the media content (Dove, p. 3, para. 0022 and 0023); create an object representation of the media content, the object representation includes the media content and fields which store attributes of the media content, the attributes includes a name of the media content and a format of the media content (Dove, p. 3, para. 0022 and 0023); remove the media content from the object representation and insert a reference to the media content into the object representation in place of the media content, then compile the object representation to byte code (Dove, p. 3, para. 0022 and 0023); create a tag header; add the media content, but not the compiled object representation, to the tag header, and transmit the compiled object representation with the compiled mark-up language description from said server to said user device for execution by said user

device to provide said particular content and said data via said user interface according to said behavior and said user interactions (Dove, p. 3, para. 0022 and 0023).

25. Claims 4, 7, 52 and 73 are rejected under 35 U.S.C. 103(a) as being unpatentable over Tuli, Davis and Dove in view of Rubin et al. (US 6,701,522 B1), hereinafter referred to as Rubin.

26. Regarding claim 4, Tuli teaches a user device including a rendering entity (col. 1, line 66 – col. 2, ll. 4, operating system with a mini-browser) but does not explicitly teach "said rendering entity is a plug-in to said browser, said plug-in is embedded in said browser before said request, and said rendering entity executes said executable code." However, in related art, Rubin teaches on this limitation wherein a portable device is equipped with plug-in installed within a user device's web browser (col. 7, ll. 18-21). One of ordinary skill in the art at the time of the applicant's invention would have found it obvious to utilize the plug-in as taught by Rubin in combination with the portable device web browser taught by Tuli. One of ordinary skill would have been motivated to combine Rubin with Tuli because plug-ins are auxiliary programs added to web browsers that provide them with new functionality (Rubin, col. 7, ll. 21-23).

27. Regarding claim 7, Tuli teaches a user device including a rendering entity (col. 1, line 66 – col. 2, ll. 4, operating system with a mini-browser) but does not explicitly teach "said rendering entity is a plug-in to said browser, said plug-in is embedded in said browser before said request, and said rendering entity executes said executable code." However, in related art, Rubin teaches on this limitation wherein a portable device is equipped with plug-in installed within a user device's web browser (col. 7, ll. 18-21).

One of ordinary skill in the art at the time of the applicant's invention would have found it obvious to utilize the plug-in as taught by Rubin in combination with the portable device web browser taught by Tuli. One of ordinary skill would have been motivated to combine Rubin with Tuli because plug-ins are auxiliary programs added to web browsers that provide them with new functionality (Rubin, col. 7, ll. 21-23).

28. Regarding claim 52, Tuli, Davis, Dove and Rubin teach a method wherein:

the server has separate object code generators and compilers for different types of rendering entities (Dove, p. 3, para. 0022, data structures);

said request is received at said server from said user device and includes an indication that identifies a type of said rendering entity from among the different types of rendering entities (Tuli, col. 2, ll. 5-13, receive request at web server); and

said compiling includes creating said executable code specific for said type of rendering entity using corresponding ones of the object code generators and compilers, in response to said indication (Dove, p. 3, para. 0022, data structures; Tuli, col. 2, ll. 5-13, mini-browser).

29. Regarding claim 73, Tuli teaches the use of a browser to display data on a user's device (col. 1, ll. 38-41) but does not explicitly teach the usage of a Flash player. Official notice is taken that it would have been obvious to one of ordinary skill in the art at the time of the applicant's invention to implement the browser to utilize a Flash player because Flash players were old and well known in the art. One of ordinary skill in the art would have been motivated to use a Flash player because of the common usage within web browsing.

30. Claims 5, 8, 51, 81, 89-97 and 101 are rejected under 35 U.S.C. 103(a) as being unpatentable over Tuli, Davis and Dove in view of Harrington (US 2002/0156909 A1).

31. Regarding claim 5, Tuli teaches the use of a browser translator to process web data (col. 1, ll. 32-36) but does not explicitly teach the compiling into ActionScript. However, in related art, Harrington teaches the common usage of ActionScript within a Flash player in a browser (p.7, para. 0055). It would have been obvious to one of ordinary skill in the art at the time of the applicant's invention to implement the browser translator to handle compilation into ActionScript because ActionScript was old and well known in the art as taught by Harrington. One of ordinary skill in the art would have been motivated to use ActionScript because of the common usage within web browsing (Harrington, p. 7, para. 0055).

32. Regarding claim 8, Tuli teaches the displaying of graphics but does not explicitly teach the browser displaying video. In related art, Harrington teaches on the aspect of displaying at least a video in the form of a Flash player utilizing ActionScript code. One of ordinary skill in the art at the time of the applicant's invention would have found it obvious to implement a video method with Harrington to enable the displaying of video content. One of ordinary skill would have been motivated because of the common usage of Flash player methods in web browsing environment for a client device as taught by Harrington (see Abstract and para. 0055).

33. Regarding claim 51, Tuli teaches the displaying of graphics but does not explicitly teach the browser displaying one of audio, video or a movie. In related art, Harrington teaches on the aspect of displaying at least a movie in the form of a Flash player

utilizing ActionScript code. One of ordinary skill in the art at the time of the applicant's invention would have found it obvious to implement a Flash method with Harrington to enable the displaying of movie content. One of ordinary skill would have been motivated because of the common usage of Flash player methods in web browsing environment for a client device as taught by Harrington (see Abstract and para. 0055).

34. Regarding claim 81, Tuli, Davis and Dove in view of Harrington teach providing an object in the executable code which identifies a name and a format of the media content, the name and format are provided via the user interface when said media content is rendered (Tuli, col. 4, ll. 18-22).

35. Regarding claims 89-97, Tuli teaches the displaying of graphics and the transcoding of the graphics content but does not explicitly teach the browser displaying one of audio, video or a movie or transcoding between MP3, WAV, MPEG, MPEG2, SORENSON, REAL, GIF, JPEG, BMP or PNG. In related art, Harrington teaches on the aspect of displaying at least a movie in the form of a Flash player utilizing ActionScript code. One of ordinary skill in the art at the time of the applicant's invention would have found it obvious to implement a Flash method with Harrington to enable the displaying of movie content. One of ordinary skill would have been motivated because of the common usage of Flash player methods in web browsing environment for a client device as taught by Harrington (see Abstract and para. 0055).

36. Regarding claim 101, Tuli teaches the displaying of graphics but does not explicitly teach the browser displaying one of audio, video or a movie. In related art, Harrington teaches on the aspect of displaying at least a movie in the form of a Flash

player utilizing ActionScript code. One of ordinary skill in the art at the time of the applicant's invention would have found it obvious to implement a Flash method with Harrington to enable the displaying of movie content. One of ordinary skill would have been motivated because of the common usage of Flash player methods in web browsing environment for a client device as taught by Harrington (see Abstract and para. 0055).

37. Claims 28, 30, 31, 62 and 80 are rejected under 35 U.S.C. 103(a) as being unpatentable over Tuli in view of Harrington and further in view of Rubin.

38. Regarding claim 28, Tuli teaches one or more processor readable storage devices having processor readable code embodied on said processor readable storage devices, said processor readable code for programming one or more processors to perform a method comprising the steps of:

receiving a request for particular content from a browser (col. 2, ll. 5-13, receive request at web server), said request is received at a server (col. 2, ll. 5-13, receive request at web server);

accessing a mark-up language description of said particular content (col. 2, lines 46-47, 57), said mark-up language description references a media file (col. 4, ll. 16-22);

compiling said mark-up language description of said particular content to create executable code (col. 2, ll. 11-12, browser translator), said executable code provides said particular content, said step of compiling is performed at said server in response to said request (col. 2, ll. 5-13, fulfill request at web server); and transmitting said executable code and said media file from said server, said media file is not compiled (col. 2, ll. 40-42, data is processed at user device).

Tuli teaches a user device including a rendering entity (col. 1, line 66 – col. 2, ll. 4, operating system with a mini-browser) but does not explicitly teach "said rendering entity is a plug-in to said browser, said plug-in is embedded in said browser before said request, and said rendering entity executes said executable code." However, in related art, Rubin teaches on this limitation wherein a portable device is equipped with plug-in installed within a user device's web browser (col. 7, ll. 18-21). One of ordinary skill in the art at the time of the applicant's invention would have found it obvious to utilize the plug-in as taught by Rubin in combination with the portable device web browser taught by Tuli. One of ordinary skill would have been motivated to combine Rubin with Tuli because plug-ins are auxiliary programs added to web browsers that provide them with new functionality (Rubin, col. 7, ll. 21-23).

Tuli teaches the displaying of graphics but does not explicitly teach the browser displaying one of audio, video or a movie. In related art, Harrington teaches on the aspect of displaying at least a movie in the form of a Flash player utilizing ActionScript code. One of ordinary skill in the art at the time of the applicant's invention would have found it obvious to implement a Flash method with Harrington to enable the displaying of movie content. One of ordinary skill would have been motivated because of the common usage of Flash player methods in web browsing environment for a client device as taught by Harrington (see Abstract and para. 0055).

39. Regarding claim 30, Tuli and Rubin teach that said executable code implements a user interface that provides access to said particular content (Tuli, col. 2, ll. 5-13, mini-browser).

40. Regarding claim 31, Tuli and Rubin teach:

said particular content includes data (Tuli, col. 2, ll. 9-11, retrieve HTML); and

said data is compiled to executable code during said step of compiling (Tuli, col. 2, ll. 9-11, browser translator).

41. Regarding claim 32, Tuli and Rubin teach that said method further comprises the steps of: transcoding said media file to an accepted format before transmitting said media file, said transcoding is separate from said compiling (Tuli, col. 2, ll. 25-29).

42. Regarding claim 62, Tuli teaches the use of a browser to display data on a user's device (col. 1, ll. 38-41) but does not explicitly teach the displaying of .SWF files. Official notice is taken that it would have been obvious to one of ordinary skill in the art at the time of the applicant's invention to implement the browser to display .SWF files because .SWF files were old and well known in the art. One of ordinary skill in the art would have been motivated to use .SWF files because of the common usage within web browsing.

43. Regarding claim 80, Tuli teaches the use of a browser to display data on a user's device (col. 1, ll. 38-41) but does not explicitly teach the usage of a Flash player. Official notice is taken that it would have been obvious to one of ordinary skill in the art at the time of the applicant's invention to implement the browser to utilize a Flash player because Flash players were old and well known in the art. One of ordinary skill in the art

would have been motivated to use a Flash player because of the common usage within web browsing.

44. Claim 11 is rejected under 35 U.S.C. 103(a) as being unpatentable over Tuli, Davis and Dove in view of Russell (2002/0069420).

45. Regarding claim 11, Tuli does not expressly disclose the step of authenticating said request, said steps of compiling and transmitting are only performed if said step of authenticating is successful, different types of authenticating are provided for different types of content or for each item of content. Russell teaches on this aspect wherein a network may authenticate a user's request to download content and that if that authentication fails, the server will not allow the user to download the content (par. 94, lines 1-10). Tuli and Russell are analogous art because they are both from the same field of endeavor of content delivery. At the time of invention it would have been obvious to a person of ordinary skill in the art to allow Tuli's invention to authenticate requests for content and to deny delivery of the content if the request does not pass authentication, as taught by Russell. The motivation for doing so would have been to ensure that the user making the request is authorized to access the content (par. 91, lines 6-7). Therefore it would have been obvious to combine Russell with Tuli and Davis for the benefit of authorized access to obtain the invention as specified in claim 11.

46. Claims 33, 41-44, 55-58, 60-62, 64, 65, 77, 78 and 98-100 are rejected under 35 U.S.C. 103(a) as being unpatentable over Tuli in view of Harrington and further in view of Rubin.

47. Regarding claim 33, Tuli teaches one or more processor readable storage devices having processor readable code embodied on said processor readable storage devices, said processor readable code for programming one or more processors to perform a method comprising the steps of:

receiving a request for particular content from a browser (col. 2, ll. 5-13, receive request at web server), said request is received at a server (col. 2, ll. 5-13, receive request at web server);

accessing a mark-up language description of said particular content (col. 2, lines 46-47, 57), said mark-up language description references a media file comprising at least one of audio, video and a movie (col. 4, ll. 16-22);

compiling said mark-up language description of said particular content to create executable code (col. 2, ll. 11-12, browser translator), said executable code provides said particular content, said step of compiling is performed at said server in response to said request (col. 2, ll. 5-13, fulfill request at web server); and

transmitting said executable code and said media file from said server, said media file is not compiled (col. 2, ll. 40-42, data is processed at user device).

Tuli teaches a user device including a rendering entity (col. 1, line 66 – col. 2, ll. 4, operating system with a mini-browser) but does not explicitly teach (a) "said rendering entity is a plug-in to said browser, said plug-in is embedded in said browser before said request, and said rendering entity executes said executable code." Tuli teaches the compilation of mark-up language for viewing by portable devices utilizing a Browser Translator (col. 2, ll. 10-12) which includes the information with respect to a view instance element that includes the tag name and attributes (col. 2, ll. 12-20, graphics and text of a normal web page) but does not specifically recite (b) the steps: "accessing source code for the view instance element, the source code includes a tag name and attributes; creating a script instruction to call an instantiation view function in a function call; adding the tag name and attributes to the function call as parameters, the instantiation view function determines, based on the parameters, what kind of objects the user device creates at a runtime of the executable code at the user device; and compiling the function call to byte code."

(a) In related art, Rubin teaches on this limitation wherein a portable device is equipped with plug-in installed within a user device's web browser (col. 7, ll. 18-21). One of ordinary skill in the art at the time of the applicant's invention would have found it obvious to utilize the plug-in as taught by Rubin in combination with the portable device web browser taught by Tuli. One of ordinary skill would have been motivated to combine Rubin with Tuli because plug-ins are auxiliary programs added to web browsers that provide them with new functionality (Rubin, col. 7, ll. 21-23).

(b) As mentioned above, Tuli teaches the compilation of mark-up language for viewing by portable devices utilizing a Browser Translator (col. 2, ll. 10-12) which includes the information with respect to a view instance element that includes the tag name and attributes (col. 2, ll. 12-20, graphics and text of a normal web page). In related art, Dove teaches the compilation of data for creating a script instruction to call an instantiation view for a function call including parameters so that the user device creates at a runtime of the executable code, compiling the function call to byte code in at least p. 3, para. 0022 and 0023. Dove teaches the creation of graphical programs that are converted to an executable format wherein the graphical programs may be represented as a plurality of data structures that define or specify the operation of the respective graphical programs. The executable format (e.g. machine language code or an interpretable script or other similar executable format) can then be executed by a portable computing device. One of ordinary skill in the art at the time of the applicant's invention would have found it obvious to implement the integration of creating an executable format file including specific function calls for graphical features to be run by a portable computing device as taught by Dove in combination with the Browser Translator taught by Tuli. One of ordinary skill would have been motivated to combine Dove with Tuli to account for the reduced processor and memory capabilities that limit the computing power of portable computing devices (Dove, p. 2, para. 0016).

48. Regarding claim 41, Tuli teaches one or more processor readable storage devices having processor readable code embodied on said processor readable storage devices, said processor readable code for programming one or more processors to perform a method comprising the steps of:

receiving a request for particular content from a browser (col. 2, ll. 5-13, receive request at web server), said request is received at a server (col. 2, ll. 5-13, receive request at web server);

accessing a mark-up language description of said particular content, said markup language description describes a behavior of said particular content on a user interface (col. 2, lines 46-47, 57), said mark-up language description references a media file (col. 4, ll. 16-22);

compiling said mark-up language description of said particular content to create executable code (col. 2, ll. 11-12, browser translator), said executable code provides said particular content, said step of compiling is performed at said server in response to said request (col. 2, ll. 5-13, fulfill request at web server); and

transmitting said executable code and said media file from said server, said media file is not compiled (col. 2, ll. 40-42, data is processed at user device).

Tuli teaches a user device including a rendering entity (col. 1, line 66 – col. 2, ll. 4, operating system with a mini-browser) but does not explicitly teach (a) "said rendering entity is a plug-in to said browser, said plug-in is embedded in said browser before said request, and said rendering entity executes said executable code." Tuli teaches the compilation of mark-up language for viewing by portable devices utilizing a Browser Translator (col. 2, ll. 10-12) which includes the information with respect to a view instance element that includes the tag name and attributes (col. 2, ll. 12-20, graphics and text of a normal web page) but does not specifically recite (b) the steps: “accessing source code for the view instance element, the source code includes a tag name and attributes; creating a script instruction to call an instantiation view function in a function call; adding the tag name and attributes to the function call as parameters, the instantiation view function determines, based on the parameters, what kind of objects the user device creates at a runtime of the executable code at the user device; and compiling the function call to byte code.”

(a) In related art, Rubin teaches on this limitation wherein a portable device is equipped with plug-in installed within a user device's web browser (col. 7, ll. 18-21). One of ordinary skill in the art at the time of the applicant's invention would have found it obvious to utilize the plug-in as taught by Rubin in combination with the portable device web browser taught by Tuli. One of ordinary skill would have been motivated to combine Rubin with Tuli because plug-ins are auxiliary programs added to web browsers that provide them with new functionality (Rubin, col. 7, ll. 21-23).

(b) As mentioned above, Tuli teaches the compilation of mark-up language for viewing by portable devices utilizing a Browser Translator (col. 2, ll. 10-12) which includes the information with respect to a view instance element that includes the tag name and attributes (col. 2, ll. 12-20, graphics and text of a normal web page). In related art, Dove teaches the compilation of data for creating a script instruction to call an instantiation view for a function call including parameters so that the user device creates at a runtime of the executable code, compiling the function call to byte code in at least p. 3, para. 0022 and 0023. Dove teaches the creation of graphical programs that are converted to an executable format wherein the graphical programs may be represented as a plurality of data structures that define or specify the operation of the respective graphical programs. The executable format (e.g. machine language code or an interpretable script or other similar executable format) can then be executed by a portable computing device. One of ordinary skill in the art at the time of the applicant's invention would have found it obvious to implement the integration of creating an executable format file including specific function calls for graphical features to be run by a portable computing device as taught by Dove in combination with the Browser Translator taught by Tuli. One of ordinary skill would have been motivated to combine Dove with Tuli to account for the reduced processor and memory capabilities that limit the computing power of portable computing devices (Dove, p. 2, para. 0016).

Tuli teaches the displaying of graphics but does not explicitly teach the browser displaying one of audio, video or a movie. In related art, Harrington teaches on the aspect of displaying at least a movie in the form of a Flash player utilizing ActionScript code. One of ordinary skill in the art at the time of the applicant's invention would have found it obvious to implement a Flash method with Harrington to enable the displaying of movie content. One of ordinary skill would have been motivated because of the common usage of Flash player methods in web browsing environment for a client device as taught by Harrington (see Abstract and para. 0055).

49. Regarding claim 43, Tuli, Harrington, Rubin and Dove teach wherein:

said particular content includes data (Tuli, col. 4, ll. 18-21); and

said data is compiled to executable code during said step of compiling (Tuli, col. 4, ll. 18-21).

50. Regarding claim 44, Tuli, Harrington, Rubin and Dove teach wherein the view instance element includes a reference to media content, and the one or more processors: access the media content (Dove, p. 3, para. 0022 and 0023); create an object representation of the media content, the object representation includes the media content and fields which store attributes of the media content, the attributes includes a name of the media content and a format of the media content (Dove, p. 3, para. 0022

and 0023); remove the media content from the object representation and insert a reference to the media content into the object representation in place of the media content, then compile the object representation to byte code (Dove, p. 3, para. 0022 and 0023); create a tag header; add the media content, but not the compiled object representation, to the tag header, and transmit the compiled object representation with the compiled mark-up language description from said server to said user device for execution by said user device to provide said particular content and said data via said user interface according to said behavior and said user interactions (Dove, p. 3, para. 0022 and 0023).

51. Regarding claim 55, Tuli, Harrington, Rubin and Dove teach the one or more processor readable storage devices wherein said mark-up language description comprises elements which are identified by markup language tags (Tuli, col. 2, ll. 9-10, use of HTML).

52. Regarding claim 56, Tuli, Harrington, Rubin and Dove teach one or more processor readable storage devices wherein: at least one of said elements define a view template of a user interface element, said view template is instantiated when said executable code is executed by said rendering entity (Tuli, col. 2, ll. 19-24).

53. Regarding claim 57, Tuli, Harrington, Rubin and Dove teach one or more processor readable storage devices wherein said elements comprise at least one element which defines a view class which supplies default properties, behavior, and child views which the view template instantiates, the child views are associated with a parent view (Tuli, col. 2, ll. 19-24).

54. Regarding claim 58, Tuli teaches the displaying of graphics but does not explicitly teach the browser displaying one of audio, video or a movie. In related art, Harrington teaches on the aspect of displaying at least a movie in the form of a Flash player utilizing ActionScript code. One of ordinary skill in the art at the time of the applicant's invention would have found it obvious to implement a Flash method with Harrington to enable the displaying of movie content. One of ordinary skill would have been motivated because of the common usage of Flash player methods in web browsing environment for a client device as taught by Harrington (see Abstract and para. 0055).

55. Regarding claim 60, Tuli, Harrington, Rubin and Dove teach at least one of said elements references a media file that contains an animation (Tuli, col. 4, ll. 16-22).

56. Regarding claim 61, Tuli teaches the displaying of graphics but does not explicitly teach the browser displaying one of audio, video or a movie. In related art, Harrington teaches on the aspect of displaying at least a movie in the form of a Flash player utilizing ActionScript code. One of ordinary skill in the art at the time of the applicant's invention would have found it obvious to implement a Flash method with Harrington to enable the displaying of movie content. One of ordinary skill would have been motivated because of the common usage of Flash player methods in web browsing environment for a client device as taught by Harrington (see Abstract and para. 0055).

57. Regarding claim 64, Tuli, Harrington, Rubin and Dove teach wherein at least one of said elements provides an inline definition of formatted text (Tuli, col. 2, ll. 59-63).

58. Regarding claim 65, Tuli, Harrington, Rubin and Dove teach wherein at least one of said elements provides an inline definition of vector graphics (Tuli, col. 2, ll. 59-63).

59. Regarding claim 77, Tuli, Harrington, Rubin and Dove teach wherein said elements comprises elements which define script code, said script code specifies a visual appearance of said user interface (Tuli, col. 2, ll. 50-55).

60. Regarding claim 78, Tuli, Harrington, Rubin and Dove teach wherein said elements comprises elements which define script code, said script code specifies an application logic of said mark-up language description (Tuli, col. 2, ll. 50-55).

61. Regarding claim 98, Tuli, Harrington, Rubin and Dove teach wherein the view instance element includes a reference to media content, and the one or more processors: access the media content (Dove, p. 3, para. 0022 and 0023); create an object representation of the media content, the object representation includes the media content and fields which store attributes of the media content, the attributes includes a name of the media content and a format of the media content (Dove, p. 3, para. 0022 and 0023); remove the media content from the object representation and insert a reference to the media content into the object representation in place of the media content, then compile the object representation to byte code (Dove, p. 3, para. 0022 and 0023); create a tag header; add the media content, but not the compiled object representation, to the tag header, and transmit the compiled object representation with the compiled mark-up language description from said server to said user device for execution by said user device to provide said particular content and said data via said user interface according to said behavior and said user interactions (Dove, p. 3, para. 0022 and 0023).

62. Regarding claim 99, Tuli, Harrington, Rubin and Dove teach the method of assembling the compiled mark-up language description and the compiled object representation into a single executable which is transmitted as said executable code from said server to said plug-in (Dove, p. 3, para. 0022).

63. Regarding claim 100, Tuli, Harrington, Rubin and Dove teach the method of transcoding said media content before the adding the media content to the tag header, said transcoding is separate from the compiling of the object (Tuli, col. 2, ll. 25-29).

64. Claims 68 and 79 is rejected under 35 U.S.C. 103(a) as being unpatentable over Tuli, Harrington, Rubin and Dove, in view of Davis.

65. Regarding claim 68, Tuli teaches the accessing of a mark-up language description but not does not explicitly teach "accessing said data at said external data source based on said one or more source files which define said connection to said external data source, said server performs said accessing." However, in related art, Davis teaches on this aspect wherein a client device can send a request to a server for secondary content (col. 5, lines 54-58) and that the secondary content can be from an external data source (abstract, line 7). Tuli and Davis are analogous art because they are both from the same field of endeavor of computer systems. At the time of invention, it would have been obvious to one of ordinary skill in the art that Davis's method of calling an application from a previously downloaded webpage could be used with Tuli's method of compiling code at a server rather than at the client. After Davis's webpage is downloaded with Tuli's system, Davis's webpage would call the secondary application and Tuli's system would then proceed to locate and compile that secondary application

for presentation to the client. The motivation for doing so would have been to allow the users of Tuli's system to be able to utilize content of the type described in Davis on a thin-client device (col. 1, ll. 14-18). Therefore it would have been obvious to combine Davis with Tuli for the benefit of utilizing more complex content on a thin-client device.

66. Regarding claim 79, Tuli teaches the accessing of a mark-up language description but not does not explicitly teach "accessing said data at said external data source based on said one or more source files which define said connection to said external data source, said server performs said accessing." However, in related art, Davis teaches on this aspect wherein a client device can send a request to a server for secondary content (col. 5, lines 54-58) and that the secondary content can be from an external data source (abstract, line 7). Tuli and Davis are analogous art because they are both from the same field of endeavor of computer systems. At the time of invention, it would have been obvious to one of ordinary skill in the art that Davis's method of calling an application from a previously downloaded webpage could be used with Tuli's method of compiling code at a server rather than at the client. After Davis's webpage is downloaded with Tuli's system, Davis's webpage would call the secondary application and Tuli's system would then proceed to locate and compile that secondary application for presentation to the client. The motivation for doing so would have been to allow the users of Tuli's system to be able to utilize content of the type described in Davis on a thin-client device (col. 1, ll. 14-18). Therefore it would have been obvious to combine Davis with Tuli for the benefit of utilizing more complex content on a thin-client device.

67. Claim 82 is rejected under 35 U.S.C. 103(a) as being unpatentable over Tuli, Davis, Dove, Harrington and further in view of Rubin.

68. Regarding claim 82, Tuli teaches a user device including a rendering entity (col. 1, line 66 – col. 2, ll. 4, operating system with a mini-browser) but does not explicitly teach "browser in a plug-in to said browser is present." However, in related art, Rubin teaches on this limitation wherein a portable device is equipped with plug-in installed within a user device's web browser (col. 7, ll. 18-21). One of ordinary skill in the art at the time of the applicant's invention would have found it obvious to utilize the plug-in as taught by Rubin in combination with the portable device web browser taught by Tuli. One of ordinary skill would have been motivated to combine Rubin with Tuli because plug-ins are auxiliary programs added to web browsers that provide them with new functionality (Rubin, col. 7, ll. 21-23).

Response to Arguments

Applicant's arguments, see Remarks, filed 18 May 2009, with respect to the rejection(s) of claim(s) 1, 5, 6, 9, 13, 21-24, 27, 28, 30-32, 37-40, 45-50, 53, 54, 67, 69, 70, 76, 81 under 35 USC 103(a) in view of Tuli (US 7,068,381) and Davis (US 6,643,696) have been fully considered and are persuasive. Therefore, the rejection has been withdrawn. However, upon further consideration, a new ground(s) of rejection is made in view of Tuli (US 7,068,381), Davis (US 6,643,696) and Dove (US 2008/0034121 A1). Applicant's arguments with respect to newly claimed subject matter are considered moot in view of the new grounds of rejection as well.

Applicant's arguments, see Remarks, filed 18 May 2009, with respect to the rejection(s) of claim(s) 4, 7, 36, 52, 55-57, 60, 62, 64, 65, 73, 77, 78, 80, 82 under 35 USC 103(a) in view of Tuli (US 7,068,381) and Davis (US 6,643,696) and Rubin (US 6,701,522) have been fully considered and are persuasive. Therefore, the rejection has been withdrawn. However, upon further consideration, a new ground(s) of rejection is made in view of Tuli (US 7,068,381), Davis (US 6,643,696), Rubin (US 6,701,522) and Dove (US 2008/0034121 A1). Applicant's arguments with respect to newly claimed subject matter are considered moot in view of the new grounds of rejection as well.

Conclusion

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Benjamin Ailes whose telephone number is (571)272-3899. The examiner can normally be reached Monday-Friday, IFP Hoteling schedule.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Andrew Caldwell can be reached on 571-272-3868. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free). If you would like assistance from a USPTO Customer Service Representative or access to the automated information system, call 800-786-9199 (IN USA OR CANADA) or 571-272-1000.

/B. A. A./
Examiner, Art Unit 2442

/Andrew Caldwell/
Supervisory Patent Examiner, Art Unit 2442